Fast Algorithm for Iris Detection

Jan Mazur

Institute of Telecommunications Teleinformatics and Acoustics,
Wroclaw University of Technology
Janiszewskiego 7/9, 50-372 Wroclaw, Poland
jan.mazur@pwr.wroc.pl

Abstract. The paper presents a fast algorithm for iris detection. The idea behind the algorithm comes from well-known Daugman's circular edge detector based on smoothed integro-differential operators. We use *binning* instead of convolution which is a bit more suitable for new generation of application processors (e.g. AP7000 from Atmel)and new CMOS imagers offering binning on demand. We also reformulated the inner function of the operator. These two steps allow for finding geometrical parameters of iris somewhat faster and with, we believe, comparable robustness offered by Daugman's solution. In the paper the idea, the implementation and some results of rough tests of the code running on AP7000 *application processor* are presented. Also, possible improvements are discussed.

Keywords: binning, iris segmentation, integro-differential operator.

1 Introduction

For many years biometrics based on iris pattern has been proving its reliability, stability and its relative robustness against spoofing. False acceptance ratio (FAR) and false rejection ratio (FRR) achieved with this method are still far lower than those achieved with other (even combined) methods of biometric identification/verification [1].

This causes that in many applications that need very high level of security iris based biometrics is considered to be very attractive and relatively cheap solution. Multi-media embedded platforms with mobile banking/trading seem to be the first non-PC solutions offering iris recognition on board. One of the first, if not the first device (PDA) on the market was Iris-SDi [8]. Also, last November OKI announced its new iris-based recognition technology and showed examples of general optical camera-equipped PDA and mobile phone that can utilize this technology [7]. For fast PCs running at 2GHz or more, efficiency of the algorithm processing an iris pattern is usually not an issue and even then manufacturers estimate the time of capturing the iris image within a range of a second and the same time range they reserve for iris identification [9].

Typical application processor used in most of mobile devices is running at a clock of only a couple of hundreds MHz and without hardware-implemented floating-point arithmetics. This makes real difficulties with efficient and relatively cheap implementation of biometric algorithms. In case of iris recognition one of the most challenging task is to localize the iris precisely and within a reasonable time. Up to now a number of quite well-performing algorithms has been proposed [2][4][5], though they are designed with the assumption that the image frame is given and it resides in memory. This is quite obvious approach with CCD imagers, although modern CMOS imagers, especially those designed as a system on chip (SOC) devices, can offer additional processing power that can be utilized by a biometric algorithm. One of the function usually offered by CMOS imagers is *binning* that, combined with windowing property, can decrease resolution dynamically to allow for faster processing, if required.

Binning can roughly be considered as a kind of joined process of convolution using flat mask and decimation, what makes it much faster than typical decimation with non-flat mask as the latter requires multiplication-summation operations and binning requires summation operation only. This is the main reason we have used binning in reformulation of the well-known Daugman's circular edge detector based on compact (not to say elegant) in form integro-differential operator [2].

In this paper we present the idea of the proposed algorithm, we try then to explain why binning is justified and what are the pros and cons of such an approach. Also, we present some preliminary results and we point out to possible improvements as well as extensions of the proposed method.

2 Proposed Solution

2.1 Statement of the Problem

For locating an iris Daugman in [2] proposes an integro-differential operator in the following form

$$\max_{(r,x_0,y_0)} \left| G_\sigma(r) * \frac{\partial}{\partial r} \oint_{r,x_0,y_0} \frac{I(x,y)}{2\pi r} ds \right| \tag{1}$$

where (r, x_0, y_0) are searched parameters (radius r and center (x_0, y_0)), $G_\sigma(r)$ is a smoothing Gaussian function of scale σ and $I(x, y)$ is an input image. The operator works as a circular edge detector, blurred at a scale defined by σ, that searches for a maximum of the given integral derivative of the input image. The whole procedure is iterative in a space spanned by all three searched parameters r, x_0, y_0 and in successively finer scales of analysis. To speed up the computations and to make the above operator more robust Daugman proposes a slight nonlinear modification and reformulates the operator so that it eventually takes the following (discrete) form

$$\max_{(n\Delta r,x_0,y_0)} \left| \sum_k \left\{ \frac{(G_\sigma(a_0) - G_\sigma(a_1)) \sum_m I(b_x, b_y)}{\Delta r \sum_m I(c_x, c_y)} \right\} \right| \tag{2}$$

with $a_0 = (n-k)\Delta r$, $a_1 = (n-k-1)\Delta r$, $b_x = k\Delta r\cos(m\Delta\phi) + x_0$, $b_y = k\Delta r\sin(m\Delta\phi) + y_0$, $c_x = (k-2)\Delta r\cos(m\Delta\phi) + x_0$, and $c_y = (k-2)\Delta r\sin(m\Delta\phi) + y_0$,

In other words for each (r, x_0, y_0) we deal with two sums over m (in numerator and denominator) and one sum over k with one multiplication in it (in numerator). In the following section we show how to modify (simplify) expression 2 to obtain a faster and, we believe, simpler algorithm that should be more easily implementable in integer arithmetics or even (partially) in assembly code, if required.

2.2 Idea of the Proposed Solution

Basically our algorithm follows the scheme proposed by Daugman in that we construct a kind of circular edge detector, though we do not use convolution as such and coarse-to-fine strategy we implement as a series of binned images, like those shown in fig. 1. Each image in the series (except the input image $I^{(0)}$) is obtained from that of higher resolution by binning the source image with a specified binning factor so that $I^{(k)} = \mathcal{B}\{I^{(k-1)}, f\}$, where f is the binning factor. In the example given in fig.1 all three binning factors defining the transitions

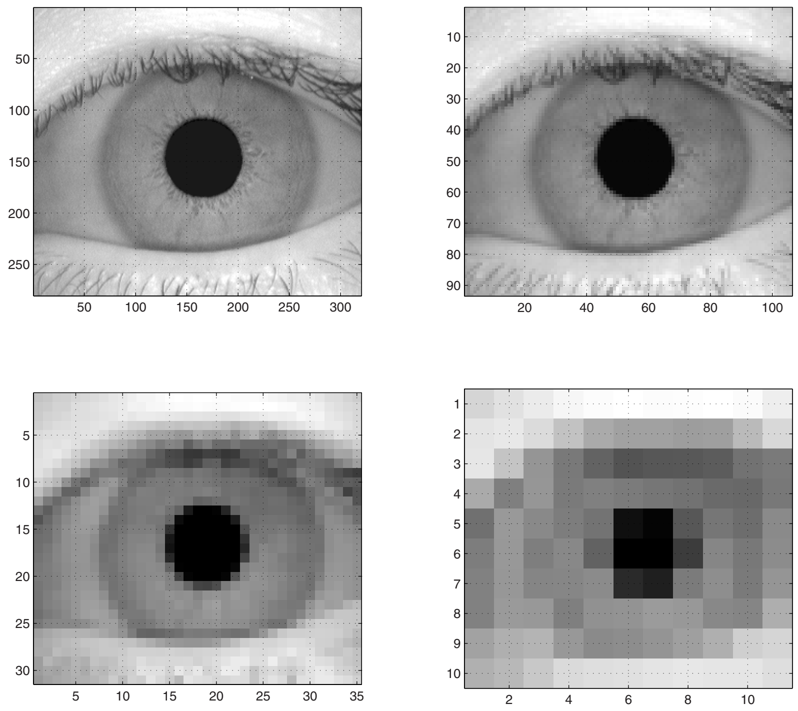

Fig. 1. An example series of binned images. The source image (upper left) comes from CASIA V1 database.

from one image to the other were set to 3, though in general their values can (or even should) be optimized for the best performance.

To implement circular edge detection we define a simple function

$$\sum_{(x,y)\in C} \langle \nabla I(x,y), \overrightarrow{r} \rangle \tag{3}$$

where $\langle .,. \rangle$ denotes inner product, $\nabla I(x,y)$ denotes the gradient of input image I at the point of analysis (x,y) and $\overrightarrow{r}$ is a vector rotating around the potential center (x_0, y_0) of the searched circle and defining the coordinates of the point of analysis as

$$x = x_m = x_0 + [r \cos \alpha_m] \tag{4}$$
$$y = y_m = y_0 + [r \sin \alpha_m] \tag{5}$$

where $\alpha_m = 2\pi m/M$ and M is the number of points taken into integration over the circle. Symbol $[\cdot]$ denotes integer value operator.

Gradient ∇I of the input image is defined by simple differences in two orthogonal directions and it is computed on demand as follows

$$\nabla I = (I_x, I_y) \tag{6}$$
$$I_x(x,y) = I(x+1,y) - I(x-1,y) \tag{7}$$
$$I_y(x,y) = I(x,y+1) - I(x,y-1) \tag{8}$$

The drawback of (3) is that it is up to the length $r = |\overrightarrow{r}|$ so that it prefers circles of larger radiuses. To avoid this problem we introduce vector $\overrightarrow{v} = \overrightarrow{r}/r$ so that $v = 1$ and the eventual version of (3) takes the form

$$\sum_{m=0}^{M-1} \langle \nabla I(x_m, y_m), \overrightarrow{v}_m \rangle \tag{9}$$

where $\overrightarrow{v}_m = (\cos \alpha_m, \sin \alpha_m)$.

The algorithm is as follows

1. Get an input image I.
2. Create a series of binned images so that $I_k = \mathcal{B}\{I_{k-1}, f\}$, $k = 1..K$
3. For I_K (i.e. the most binned image in the series) find a minimum and assume it is a center $(x_0^{(K)}, y_0^{(K)})$ of the pupil. This step may require smoothing with a small (3x3) Gaussian mask. Set $r^{(K)} = 2$; this is true for all CASIA images and in case the size of the most binned image is about 10 x 10 pixels. In other cases this value should be adjusted.
4. For each image $I^{(k)}$ $k = 0..K$ starting from K to 0 (i.e. from more to less binned images)

(a) construct a set of potential centers around the point defined by initial values obtained as the result of the previous stage

$$(x_0^{(k)}, y_0^{(k)}) \in C^{(k)} \tag{10}$$

where

$$C^{(k)} = \{(\hat{x}_0^{(k)} - [f_k/2])..(\hat{x}_0^{(k)} + [f_k/2])\} \times \{(\hat{y}_0^{(k)} - [f_k/2])..(\hat{y}_0^{(k)} + [f_k/2])\} \tag{11}$$

(b) construct a set of potential radiuses

$$r^{(k)} \in \{(\hat{r}^{(k)} - [f_k/2])..(\hat{r}^{(k)} + [f_k/2])\} \tag{12}$$

(c) find (we omit the upper k index for editorial purpose)

$$(\check{r}, \check{x}, \check{y}) = \arg \max_{(r,x,y)} \sum_{m=1}^{M} (I_x(x_m, y_m) \cos \alpha_m + I_y(x_m, y_m) \sin \alpha_m) \tag{13}$$

5. Recompute $(\check{r}^{(k)}, \check{x}^{(k)}, \check{y}^{(k)})$ to the finer grid using f_k

$$(\hat{r}^{(k-1)}, \hat{x}^{(k-1)}, \hat{y}^{(k-1)}) = f_k(\check{r}^{(k)}, \check{x}^{(k)}, \check{y}^{(k)}) \tag{14}$$

In much the same way we search for external radius of an iris and for the center if required.

In the following sections we present and discuss the results of some rough tests and we point out to possible enhancements of the proposed method.

3 Results

The algorithm presented in the previous section has been implemented in C and it runs on AP7000, a 133MHz application processor.

We have tested the algorithm using CASIA V1 and V3 database[1][6]. The results of some preliminary tests are summarized in table 1. We assumed $M = 64$ (i.e. the number of points of integration) at all stages, but for $M = 32$ the quality seems to stay the same, so the real times can possibly be approximately half of those shown in the table. The results given concern inner and outer circle and their respective centers; they do not cover other processing sometimes performed like eyelid detection.

The most important parameter in the proposed algorithm to be set up is the area of potential centers of pupil. In the tests performed we assumed that the actual center of pupil is covered by the area of the darkest pixel in the frame of the lowest resolution (in our case: L_3). In higher resolution frame (one level up) this area is covered by a number of pixels that is up to the relevant binning factor. In our case this factor equals 3, hence nine pixels (a 3×3 matrix treated as a central

[1] Portions of the research in this paper use the CASIA-IrisV1 and CASIA-IrisV3 collected by the Chinese Academy of Sciences Institute of Automation (CASIA).

Table 1. Execution times obtained for three cases C_1, C_2 and C_3 (see text for explanation). Tk denotes the time required for computations on k-th level of binning.

	C_1	C_2	C_3
Binning	50ms	50ms	50ms
T1+T2+T3	10ms	20ms	30ms
T0	10ms	20ms	30ms

point and the neighborhood of ±1 pixel) in the frame L_2 matches the area of one pixel in L_3. This matrix is the set of potential centers of the pupil. At the same time we set the potential range of pupil's radius and for these three parameters the algorithm finds the best solution. Having found the new coordinates of the center of pupil and new radius, we iterate this scheme until we will find the searched parameters in the frame of highest resolution. Theoretically, it should be enough to take into account only those pixels in L_i that cover the area of exactly one pixel in L_{i+1}. Unfortunately, this is often not true because pixels in lower resolution are greatly affected by their respective neighbors and hence the area of potential centers in higher resolution frames should be generally larger than the area covered by one pixel in the lower resolution frame.

We considered the following three cases:

C1 - neighborhood of ±1 pixel for all stages,
C2 - neighborhood of ±2 pixel for stages L1 and L2
C3 - neighborhood of ±2 pixel for all stages

The times given in table 1 are very roughly estimated, especially for small values, as the program was running under linux and the least step of clock() function used to measure the execution time was 10ms.

In fig. 2a we show an example of image with found circles drawn. In fig. 2b we show the zoomed version of the image from the left with visible path of centers recomputed to the highest resolution (i.e. the original resolution of the image). In the given example centers of both inner and outer circle of iris were searched. It is, however possible to assume that pupil and iris are coaxial and finding of center of outer circle can be omitted. This, of course, reduces the total time of computations.

Two main issues have been addressed in the tests we have performed. One of the issues was the execution time. The other issue was the accuracy with which parameters are estimated. In the following figures we try to summarize the behavior of the proposed algorithm. In fig. 3 we show correct estimation of parameters in case of white noise added to the images. The proposed algorithm is relatively immune to additive noise. This feature can be important when working in low light conditions.

On the other hand, there is a number of clear images (i.e. with no noise added) for which the proposed algorithm found the parameters incorrectly. In the rough tests made on more than 400 images taken from CASIA V3-INT database about 20%-25% of images have been processed incorrectly. For these images the inner

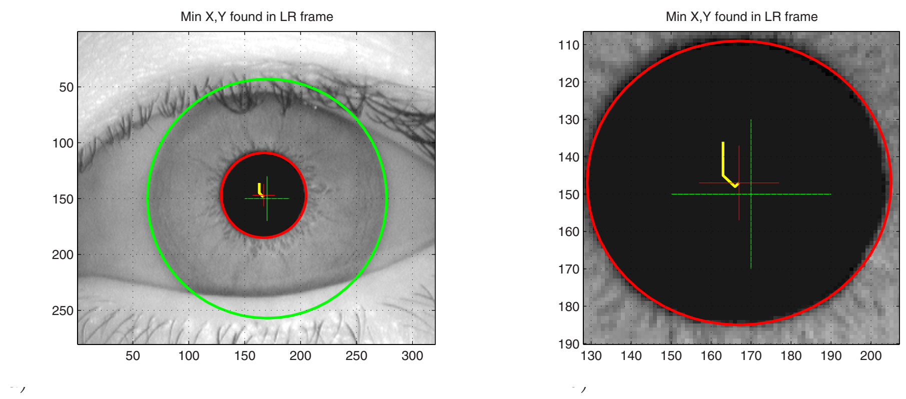

Fig. 2. An example of image with the inner and outer boundary drawn (a) and inner and outer centers drawn (b). The light curve shows the path of centers obtained in each of 4 stages of processing.

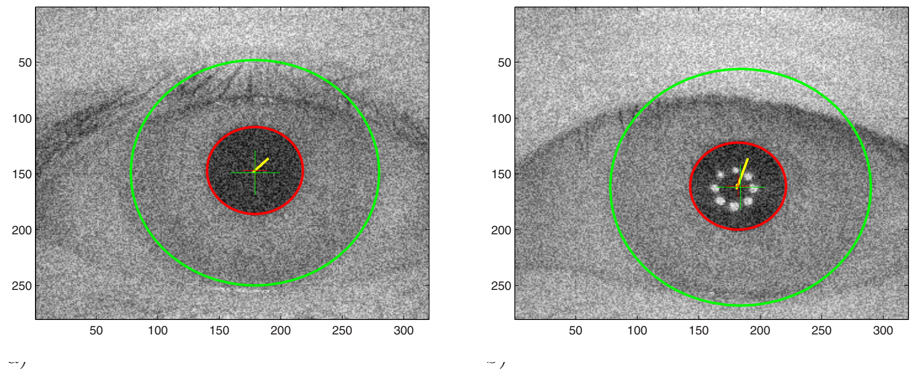

Fig. 3. An example of images from: a) CASIA V1 database b) CASIA V3-INT database with additive noise

or outer radius of the iris and/or centers of pupil have been found incorrectly. In fig. 4a the center of the pupil and pupil's radius have been found incorrectly. The initial guess of center coordinates is here affected by two factors: first, too bright pupil area due to existing reflections and second, relatively dark area of eyelashes. This moves the found center to the upper left part of the pupil and because of neighborhood values taken into account, this is too far from the real center of the pupil and the algorithm finds the best pupil's radius but only in the given range. The center of outer circle is only allowed to move a couple of pixels from the center of inner circle, so it also finds the radius incorrectly as the best solution in the given range. Note that in the proposed method initial estimation of pupil's center is critical. Due to reasonable constraints all parameters will be found incorrectly, if pupil's center can not be estimated with enough accuracy.

In fig. 4b only the inner radius of the iris has been found incorrectly. This is because in the mean sense (as we integrate) the set of lamps reflected from the pupil can be considered as relatively sharp edge; sharper than any other in the considered set of potential centers and radiuses. Note, that the real edge of the pupil can be even sharper but due to assumed constraints concerning the set of potential centers and the range of potential radiuses chosen in each frame of different resolutions, the one that was found is the best solution. To overcome this kind of errors we could set the minimal value of inner radius to be large enough or we could decrease the intensity of the reflection of lamps. In general, significant number of all the errors came from the fact that pupil reflecting the lamps could not be assumed to be the darkest area in the given image. Also, some nonlinear mappings of intensity could help to reduce the number of errors coming from reflections.

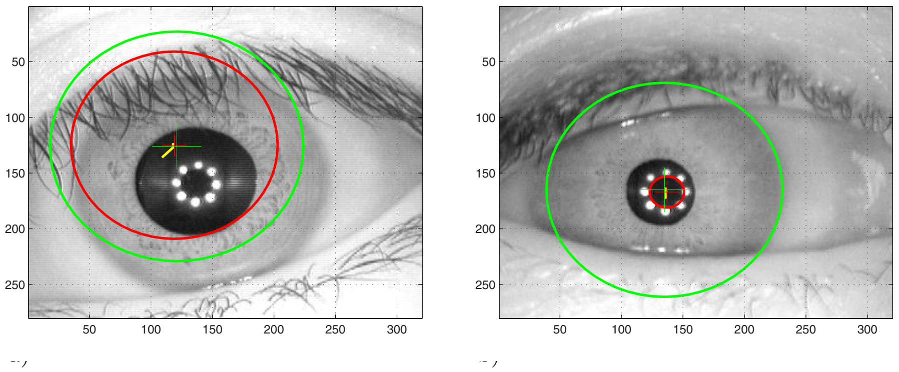

Fig. 4. An example of incorrect estimation of: a) the center of the pupil and - implicitly - the rest of parameters, b) inner radius of an iris,.

The tests concerning part of CASIA V3-LAMP database resulted in a slight smaller error rates of about 15%-20%. A reason for that is the reflections in images of CASIA V3-Lamp occupy relatively smaller area than reflections in CASIA-V3 int. An example of correct estimation of parameters from CASIA V3-Lamp and CASIA V3-Int has been shown in fig. 5a and fig. 5b, respectively.

4 Discussion

The results presented in table 1 shows that the most computationally intensive is binning. What can be done here to decrease the time of computations is to analyze the code and optimize it taking into account that in case of binning we basically run (almost) empty loops. The other possibility is to interface the imager and the processor with a small FPGA or even CPLD device utilizing its capability of running parallel computations. Also, application processors usually have some kind of hardware-implemented accelerator for fixed point or integer arithmetics that can be used to speed up the algorithm.

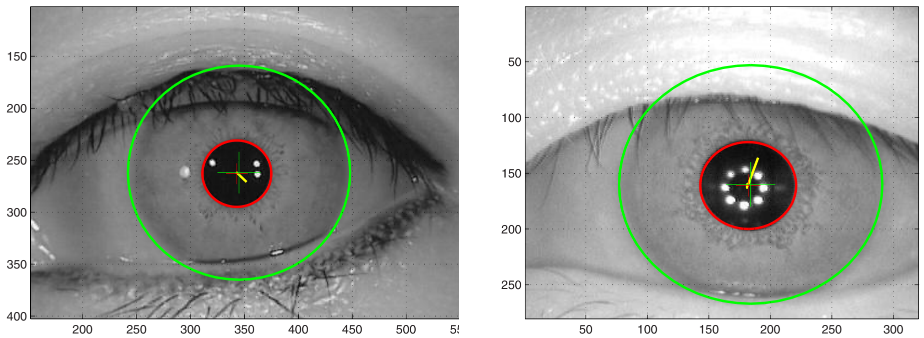

Fig. 5. An example of correctly estimated parameters of iris for image from: a) CASIA V3-LAMP, b) CASIA V3-INT (a noisy version of this image is shown in fig. 3b)

From algorithmic point of view, the proposed solution requires many optimizations, including the choice of binning factors and, implicitly, the number of stages, the number (M) of points on the circle taken for integration and the area of potential centers as well as the range of radiuses that are considered to find the best solution. Undersampling is also possible, especially in higher resolutions.

Further, a new method of aquisition can be employed, that could be called "track and get". With such a method we could track an iris in very low resolutions (quickly) until it is in an appropriate position and then using the information on that position we could take just part of one (windowing property of CMOS imagers), high resolution frame. This would reduce the number of binning stages and consequently the time required for making unneeded binned images. This would also reduce the time required for finding searched parameters. In effect, doing so, we would virtually move a part of the algorithm to the imager equipped with binning capabilities.

5 Conclusion

In the paper we have presented a fast algorithm for iris detection or, more precisely, for finding geometrical parameters of an iris. Basically the algorithm is a reformulation of Daugman's circular detector based on integro-differential operator [2]. The idea behind the proposed method comes from technology and it tries to utilize features of modern imagers usually equipped with so called *binning*, though it does not require an imager to have this function implemented. We also proposed a relatively simple measure of circularity that can be easily implemented in assembly code, if required.

The results obtained showed that initial guess of pupil's center (in the lowest resolution considered) is critical for the algorithm to work properly. The results also showed that a number of parameters should be adjusted and their values are up to the resolution of the imager, processor that is to be used and the required efficiency. Optimizing those parameters, possibly utilizing the parallelism offered by small FPGA or CPLD device, should make the proposed algorithm fast enough to be implemented in small, cost effective mobile devices. The eventual time required for finding geometrical parameters of an iris with the use of the (optimized) method proposed and the optimized code should be no more than 100ms for a typical 1.3-2.0 Mpix imager.

References

1. Jain, A., Bolle, R., Pankanti, S. (eds.): Biometrics: Personal Identification in a Networked Society. Kluwer, Norwell (1999)
2. Daugman, J.G.: High Confidence Visual Recognition of Persons by a Test of Statistical Independence. IEEE Trans. Pattern Analysis and Machine Intelligence 15(11), 1148–1161 (1993)
3. Daugman, J.G.: The importance of being random: statistical principles of iris recognition. Pattern Recognition 36(2), 279–291
4. Cho, D.H., Park, K.R., Rhee, D.W.: Real-Time Iris Localization for Iris Recognition in Cellular Phone, snpd-sawn. In: Sixth Int. Conf. on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS Int. Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05), pp. 254–259 (2005)
5. Feng, X., Fang, C., Ding, X., Wu, Y.: Iris Localization with Dual Coarse-to-fine Strategy. In: icpr.18h Int. Conf. on Pattern Recognition (ICPR'06), pp. 553–556 (2006)
6. `http://www.cbsr.ia.ac.cn/IrisDatabase.htm`
7. `http://www.oki.com/en/press/2006/z06114e.html`
8. `http://www.iridiantech.com/products.php?page=4`
9. `http://www.oki.com/jp/FSC/iris/en/m_features.html`